```
ISIS-II PL/M-80 V3.1 COMPILATION OF MODULE HELP
OBJECT MODULE PLACED IN HELP.OBJ
COMPILER INVOKED BY:  PLM80 HELP.PLM DEBUG OPTIMIZE
```

```
                $title ('Help Utility Version 1.1')
  1             help:
                do;

                /*
                  Copyright (C) 1982
                  Digital Research
                  P.O. 579
                  Pacific Grove, CA 93950

                  Revised:
                    06 Dec 82  by Bruce Skidmore
                */

  2   1           declare plm label public;

                /***********************************************
                        Interface Procedures
                ***********************************************/
  3   1           mon1:
                  procedure (func,info) external;
  4   2             declare func byte;
  5   2             declare info address;
  6   2           end mon1;

  7   1           mon2:
                  procedure (func,info) byte external;
  8   2             declare func byte;
  9   2             declare info address;
 10   2           end mon2;

 11   1           mon3:
                  procedure (func,info) address external;
 12   2             declare func byte;
 13   2             declare info address;
 14   2           end mon3;

                /***********************************************
                         Global Variables
                ***********************************************/

 15   1           declare (list$mode,nopage$mode,create$mode,extract$mode,page$mode) byte;
 16   1           declare (offset,eod) byte;

 17   1           declare fcb (13) byte external;
 18   1           declare fcb2 (36) byte;

 19   1           declare maxb address external;
 20   1           declare fcb16 (1) byte external;
 21   1           declare tbuff (128) byte external;
```

CP/M Plus Ver 3.0
COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. # CP3-135

```
 22  1        declare control$z literally '1AH';
 23  1        declare cr literally '0DH';
 24  1        declare lf literally '0AH';
 25  1        declare tab literally '09H';
 26  1        declare slash literally '''/''';
 27  1        declare true literally '0FFH';
 28  1        declare false literally '00H';

 29  1        declare (cnt,index) byte;
 30  1        declare sub(12) byte;
 31  1        declare com(11) structure(
                name(15) byte);

 32  1        declare sysbuff(8) structure(
                subject(12) byte,
                record address
                rec$offset byte,
                level byte) at (.memory);

 33  1      declare name(12) byte;
 34  1      declare level byte;
 35  1      declare gindex address;
 36  1      declare tcnt byte;
 37  1      declare version address;
 38  1      declare page$len byte;
 39  1      declare display$cols byte;
 40  1      declare clear$screen (26) byte initial (cr,lf,lf,lf,lf,lf,lf,
                                                    lf,lf,lf,lf,lf,lf,
                                                    lf,lf,lf,lf,lf,lf,
                                                    lf,lf,lf,lf,lf,lf,'$');

              /**************************************
               *                                    *
               *      B D O S   Externals           *
               *                                    *
               **************************************/

 41  1        read$console:
                procedure byte;
 42  2            return mon2 (1,0);
 43  2          end read$console;

 44  1        write$console:
                procedure (char);
 45  2            declare char byte;
 46  2            call mon1 (2,char);
 47  2          end write$console;

 48  1        print$console$buf:
                procedure (buff$adr);
 49  2            declare buff$adr address;
 50  2            call mon1 (9,buff$adr);
 51  2          end print$console$buf;

 52  1        read$console$buff:
                procedure (buff$adr);
 53  2            declare buff$adr address;
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #

```
 54  2          call mon1(10,buff$adr);
 55  2        end read$console$buff;

 56  1      direct$con$io:
              procedure(func) byte;
 57  2          declare func byte;
 58  2            return mon2(6,func);
 59  2        end direct$con$io;

 60  1      get$version:
              procedure  address;
 61  2          return mon3(12,0);
 62  2        end get$version;

 63  1      delete$file:
              procedure (fcb$address);
 64  2          declare fcb$address address;
 65  2          call mon1(19,fcb$address);
 66  2        end delete$file;

 67  1      open$file:
              procedure (fcb$address) byte;
 68  2          declare fcb$address address;
 69  2          declare fcb based fcb$address (1) byte;
 70  2          fcb(12) = 0; /* EX = 0 */
 71  2          fcb(32) = 0; /* CR = 0 */
 72  2          return mon2 (15,fcb$address);
 73  2        end open$file;

 74  1      close$file:
              procedure (fcb$address) byte;
 75  2          declare fcb$address address;
 76  2          return mon2 (16,fcb$address);
 77  2        end close$file;

 78  1      read$record:
              procedure (fcb$address) byte;
 79  2          declare fcb$address address;
 80  2          return mon2 (20,fcb$address);
 81  2        end read$record;

 82  1      write$record:
              procedure (fcb$address) byte;
 83  2          declare fcb$address address;
 84  2          return mon2(21,fcb$address);
 85  2        end write$record;

 86  1      make$file:
              procedure (fcb$address) byte;
 87  2          declare fcb$address address;
 88  2          declare fcb based fcb$address (1) byte;
 89  2          fcb(12) = 0; /* EX = 0 */
 90  2          fcb(32) = 0; /* CR = 0 */
 91  2          return mon2(22,fcb$address);
 92  2        end make$file;

 93  1      read$rand:
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950

SER. #_____

```
                    procedure (fcb$address) byte;
 94   2                    declare fcb$address address;
 95   2                    return mon2(33,fcb$address);
 96   2                  end read$rand;

 97   1                set$dma:
                          procedure (dma$address);
 98   2                    declare dma$address address;
 99   2                    call mon1(26,dma$address);
100   2                  end set$dma;

101   1                set$rand$rec:
                          procedure (fcb$address);
102   2                    declare fcb$address address;
103   2                    call mon1(36,fcb$address);
104   2                  end set$rand$rec;

105   1                terminate:
                          procedure;
106   2                    call mon1 (0,0);
107   2                  end terminate;

                    /**********************************************
                                Error Procedure

                          Displays error messages and
                          terminates if required.
                    **********************************************/
108   1                error:
                          procedure(term$code,err$msg$adr);
109   2                    declare term$code byte;
110   2                    declare err$msg$adr address;

111   2                    call print$console$buf(.(cr,lf,'ERROR:  $'));
112   2                    call print$console$buf(err$msg$adr);
113   2                    call print$console$buf(.(cr,lf,'$'));
114   2                    if term$code then
115   2                      call terminate;
116   2                  end error;

                    /**********************************************
                                Move Procedure

                          Moves specified number of bytes
                          from the Source address to the
                          Destination address.
                    **********************************************/
117   1                movef:
                          procedure (mvcnt,source$addr,dest$addr);
118   2                    declare (source$addr,dest$addr) address;
119   2                    declare mvcnt byte;
120   2                    call move(mvcnt,source$addr,dest$addr);
121   2                    return;
122   2                  end movef;

                    /**********************************************
                                Compare Function
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #

```
                    Compares 12 byte strings

                    Results:  0 - string1 = string2
                              1 - string1 < string2
                              2 - string1 > string2
                ***********************************************/
123   1         compare:
                  procedure(str1$addr,str2$addr) byte;
124   2             declare (str1$addr,str2$addr) address;
125   2             declare string1 based str1$addr (12) byte;
126   2             declare string2 based str2$addr (12) byte;
127   2             declare (result,i) byte;
128   2             result,
                    i = 0;
129   2             do while (i < 12) and (string1(i) <> ' ');
130   3                if string1(i) <> string2(i) then
131   3                do;
132   4                   if string1(i) < string2(i) then
133   4                   do;
134   5                      result = 1;
135   5                   end;
                          else
136   4                   do;
137   5                      result = 2;
138   5                   end;
139   4                   i = 11;
140   4                end;
141   3                i = i + 1;
142   3             end;
143   2             return result;
144   2         end compare;

                /***********************************************
                        Increment Procedure

                   Increments through a record.
                ***********************************************/
145   1         inc:
                  procedure (inci) byte;
146   2             declare inci byte;
147   2             inci = inci + 1;
148   2             if inci > 127 then
149   2             do;
150   3                if read$record(.fcb) = 0 then
151   3                do;
152   4                   inci = 0;
153   4                end;
                          else
154   3                do;
155   4                   eod = true;
156   4                   inci = 0;
157   4                end;
158   3             end;
159   2             return inci;
160   2         end inc;
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #

```
                /**************************************
                      Page$check Procedure

                   Halts display after page$len lines
                **************************************/
161   1         page$check:
                  procedure(line$cnt$addr) byte;
162   2            declare line$cnt$addr address;
163   2            declare line$cnt based line$cnt$addr byte;
164   2            declare quit byte;
165   2            quit = 0;
166   2            if (not nopage$mode) and (page$mode) then
167   2            do;
168   3               if (line$cnt:=line$cnt+1) > page$len then
169   3               do;
170   4                  call print$console$buf(.(cr,lf,'Press RETURN to continue $'));
171   4                  line$cnt = 0;
172   4                  do while (line$cnt = 0);
173   5                     line$cnt = direct$con$io(0FDH);
174   5                  end;
175   4                  call print$console$buf(.(cr,'                              ',
                                                                          cr,'$'));
176   4                  if line$cnt = 3 /* control c */ then
177   4                  do;
178   5                     line$cnt = close$file(.fcb);
179   5                     call terminate;
180   5                  end;
                           else
181   4                  do;
182   5                     if line$cnt <> cr then
183   5                     do;
184   6                        quit = true;
185   6                     end;
186   5                     line$cnt = 0;
187   5                  end;
188   4               end;
                        else
189   3               do;
190   4                  call write$console(lf);
191   4               end;
192   3            end;
                   else
193   2            do;
194   3               line$cnt = 0;
195   3               call write$console(lf);
196   3            end;
197   2            return quit;
198   2         end page$check;

                /**********************************************
                            Init Procedure

                      Reads the index into memory
                **********************************************/
199   1         init:
                  procedure;
200   2            declare (buf$size,max$buf,init$i) address;
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950
SER. #_____

```
201   2          declare end$index byte;
202   2          buf$size = maxb - .memory;
203   2          max$buf = buf$size;
204   2          end$index = 0;
205   2          init$i = 7;
206   2          do while (not end$index) and (max$buf > 127);
207   3            call set$dma(.sysbuff(init$i-7).subject);
208   3            if read$record(.fcb) <> 0 then
209   3            do;
210   4              init$i = close$file(.fcb);
211   4              call error(true,.('Reading HELP.HLP index.$'));
212   4            end;
213   3            if sysbuff(init$i).subject(0) = '$' then end$index = true;
215   3            if not end$index then
216   3            do;
217   4              max$buf = max$buf - 128;
218   4              init$i = init$i + 8;
219   4            end;
220   3          end;
221   2          call set$dma(.tbuff);
222   2          if (max$buf < 128) and (not end$index) then
223   2          do;
224   3            init$i = close$file(.fcb);
225   3            call error(true,.('Too many entries in Index Table.',
                            '  Not enough memory.$'));
226   3          end;
227   2        end init;


          /*********************************************
                      Parse Procedure

               Parses the command tail
          *********************************************/
228   1     parse:
          procedure byte;
229   2       declare (index,begin,cnt,i,stop,bracket) byte;
230   2       index = 0;
231   2       if tbuff(0) <> 0 then
232   2       do;
233   3         do index = 1 to tbuff(0);
234   4           if tbuff(index) = tab then tbuff(index) = ' ';
236   4           else if tbuff(index) = ',' then tbuff(index) = ' ';
                  end;
239   3         index = 1;
240   3         do while(index < tbuff(0)) and (tbuff(index) = ' ');
241   4           index = index + 1;
242   4         end;
243   3         if tbuff(index) = ',' then
244   3           do;
245   4             begin = level;
246   4             tbuff(index) = ' ';
247   4           end;
                  else
248   3             begin = 0;
249   3           do index = begin to 10;
250   4             call movef(15,.('              ',cr,'$'),.com(index).name);
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #_____

```
251  4          end;
252  3          index = begin;
253  3          cnt = 1;
254  3          stop,
                bracket = 0;
255  3          do while (tbuff(cnt) <> 0) and (not stop);
256  4             if (tbuff(cnt) <> 20H) then
257  4             do;
258  5                i = 0;
259  5                do while (((tbuff(cnt) <> 20H) and (tbuff(cnt) <> '[')) and
                              (tbuff(cnt) <> 0)) and ((i < 12) and (index < 11));
260  6                   if (tbuff(cnt) > 60H) and (tbuff(cnt) < 7BH) then
261  6                   do;
262  7                      com(index).name(i) = tbuff(cnt) - 20H;
263  7                   end;
                         else
264  6                   do;
265  7                      com(index).name(i) = tbuff(cnt);
266  7                   end;
267  6                   cnt = cnt + 1;
268  6                   i = i + 1;
269  6                end;
270  5                index = index + 1;
271  5                if (bracket or (index > 10)) then
272  5                do;
273  6                   stop = true;
274  6                end;
                         else
275  5                if tbuff(cnt) = '[' then
276  5                do;
277  6                   if com(index-1).name(0) = ' ' then index = index - 1;
279  6                   com(index).name(0) = '[';
280  6                   cnt = cnt + 1;
281  6                   index = index + 1;
282  6                   bracket = true;
283  6                end;
                      end;
                      else
285  4             do;
286  5                cnt = cnt + 1;
287  5             end;
288  4          end;
289  3       end;
290  2       list$mode,
                nopage$mode,
                create$mode,
                extract$mode = false;
291  2       if index > 0 then
292  2       do;
293  3          i = 0;
294  3          do while (i < 10);
295  4             if com(i).name(0) = '[' then
296  4             do;
297  5                if (com(i+1).name(0) = 'C') then
298  5                do;
299  6                   create$mode = true;
300  6                   index = index - 2;
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #

```
301  6                end;
302  5                else if (com(i+1).name(0) = 'E') then
303  5                     do;
304  6                        extract$mode = true;
305  6                        index = index - 2;
306  6                     end;
307  5                     else if (com(i+1).name(0) = 'N') then
308  5                          do;
309  6                             nopage$mode =true;
310  6                             index = index - 2;
311  6                          end;
312  5                          else if (com(i+1).name(0) = 'L') then
313  5                               do;
314  6                                  list$mode = true;
315  6                                  nopage$mode = true;
316  6                                  index = index - 2;
317  6                               end;
318  5                               else if (com(i+1).name(0) <> ' ') then
319  5                                    do;
320  6                                      index = index - 2;
321  6                                    end;
                                          else
322  5                                         do;
323  6                                         index = index - 1;
324  6                                         end;
325  5              i = 10;
326  5            end;
327  4            i = i + 1;
328  4         end;
329  3      end;
330  2      return index;
331  2    end parse;

           /*****************************************
                 Create$index Procedure

                Creates HELP.HLP from HELP.DAT
           *****************************************/
332  1    create$index:
       procedure;
333  2      declare (cnt, i, rec$cnt) byte;
334  2      declare (index,count,count2,max$buf,save$size) address;
335  2      declare fcb3(36) byte;
336  2      call print$console$buf(.(cr,lf,'Creating HELP.HLP...$'));
337  2      do i = 0 to 7;
338  3         call movef(12,.('$          '),.sysbuff(i).subject);
339  3      end;
340  2      rec$cnt,
           index = 0;
341  2      save$size = maxb - .memory;
342  2      max$buf = save$size;
343  2      call movef(13,.(0,'HELP    DAT',0),.fcb);
344  2      if open$file(.fcb) = 0FFH then
345  2      do;
346  3         call error(true,.('HELP.DAT not on current drive.$'));
347  3      end;
348  2      eod = 0;
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950

SER. #

```
349  2          do while (not eod) and (read$record(.fcb) = 0);
350  3             i = 0;
351  3             do while(i < 128) and (not eod);
352  4                if tbuff(i) = control$z then
353  4                do;
354  5                   eod = true;
355  5                end;
                      else
356  4                do;
357  5                   if tbuff(i) = slash then
358  5                   do;
359  6                      cnt = 0;
360  6                      do while(not eod) and (tbuff(i) = slash);
361  7                         i = inc(i);
362  7                         cnt = cnt + 1;
363  7                      end;
364  6                      if (cnt = 3) and (not eod) then
365  6                      do;
366  7                         sysbuff(index).level = tbuff(i) - '0';
367  7                         i = inc(i);
368  7                         cnt = 0;
369  7                         do while ((cnt < 12) and (not eod)) and (tbuff(i) <> cr);
370  8                            if (tbuff(i) > 60H) and (tbuff(i) < 7BH) then
371  8                            do;
372  9                               sysbuff(index).subject(cnt) = tbuff(i) - 20H;
373  9                            end;
                                  else
374  8                            do;
375  9                               sysbuff(index).subject(cnt) = tbuff(i);
376  9                            end;
377  8                            i = inc(i);
378  8                            cnt = cnt + 1;
379  8                         end;
380  7                         if (not eod) then
381  7                         do;
382  8                            call set$rand$rec(.fcb);
383  8                            call movef(1,.fcb(33),.sysbuff(index).record);
384  8                            call movef(1,.fcb(34),.sysbuff(index).record+1);
385  8                            sysbuff(index).record = sysbuff(index).record - 0001H;
386  8                            sysbuff(index).rec$offset = i;
387  8                            index = index + 1;
388  8                            if ((index mod 8) = 0) then
389  8                            do;
390  9                               rec$cnt = rec$cnt + 1;
391  9                               max$buf = max$buf - 128;
392  9                               if (max$buf < 128) and (not eod) then
393  9                               do;
394 10                                  cnt = close$file(.fcb);
395 10                                  call error(true,
                                                .('Too many entries in Index Table.',
                                                  ' Not enough memory.$'));
396 10                               end;
                                     else
397  9                               do count = index to index + 7;
398 10                                  call movef(12,.('$          '),
                                                .sysbuff(count).subject);
399 10                               end;
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950

SER. #_____

```
400  9                    end;
401  8                  end;
402  7                end;
403  6              end;
                    else
404  5              do;
405  6                i = inc(i);
406  6              end;
407  5            end;
408  4          end;
409  3        end;
410  2        call set$dma(.sysbuff);
411  2        rec$cnt = rec$cnt + 1;
              /**********************************
                  create HELP.HLP
              **********************************/
412  2        call movef(13,.(0,'HELP    HLP',0),.fcb3);
413  2        call delete$file(.fcb3);
414  2        if make$file(.fcb3) = 0FFH then
415  2        do;
416  3          cnt = close$file(.fcb2);
417  3          call delete$file(.fcb2);
418  3          cnt = close$file(.fcb);
419  3          call error(true,.('Unable to Make HELP.HLP.$'));
420  3        end;
421  2        call movef(4,.(0,0,0,0),.fcb2+32);
422  2        cnt = read$rand(.fcb2);
423  2        do count = 0 to index - 1;
424  3          sysbuff(count).record = sysbuff(count).record + rec$cnt;
425  3        end;
426  2        do count = 0 to rec$cnt - 1;
427  3          call set$dma(.memory(shl(count,7)));
428  3          if write$record(.fcb3) = 0FFH then
429  3          do;
430  4            cnt = close$file(.fcb3);
431  4            call delete$file(.fcb3);
432  4            cnt = close$file(.fcb2);
433  4            call delete$file(.fcb2);
434  4            cnt = close$file(.fcb);
435  4            call error(true,.('Writing file HELP.HLP.$'));
436  4          end;
437  3        end;
438  2        call movef(4,.(0,0,0,0),.fcb+32);
439  2        cnt = read$rand(.fcb);
440  2        eod = 0;
441  2        do while (not eod);
442  3            count = 0;
443  3            max$buf = save$size;
444  3            do while (not eod) and (max$buf > 127);
445  4              call set$dma(.memory(shl(count,7)));
446  4              if read$record(.fcb) <> 0 then
447  4              do;
448  5                eod = true;
449  5              end;
                    else
450  4              do;
451  5                max$buf = max$buf - 128;
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #

```
452  5                  count = count + 1;
453  5                end;
454  4              end;
455  3              do count2 = 0 to count-1;
456  4                 call set$dma(.memory(shl(count2,7)));
457  4                 if write$record(.fcb3) = 0FFH then
458  4                 do;
459  5                    i = close$file(.fcb3);
460  5                    call delete$file(.fcb3);
461  5                    i = close$file(.fcb);
462  5                    call error(true,.('Writing file HELP.HLP.$'));
463  5                 end;
464  4              end;
465  3           end;
466  2           if close$file(.fcb) = 0FFH then
467  2           do;
468  3              cnt = close$file(.fcb3);
469  3              call error(true,.('Closing file HELP.DAT.$'));
470  3           end;
471  2           if close$file(.fcb3) = 0FFH then
472  2           do;
473  3              call error(true,.(false,'Closing file HELP.HLP.$'));
474  3           end;
475  2           call print$console$buf(.('HELP.HLP created',cr,lf,'$'));
476  2         end create$index;

           /*********************************************
                  Extract$file Procedure

               Creates HELP.DAT from HELP.HLP
           *********************************************/
477  1       extract$file:
             procedure;
478  2           declare (end$index,i) byte;
479  2           declare (count,count2,max$buf,save$size) address;

480  2            call print$console$buf(.(cr,lf,'Extracting data....$'));
481  2            call movef(13,.(0,'HELP    HLP',0),.fcb);
482  2            if open$file(.fcb) = 0FFH then
483  2            do;
484  3               call error(true,.('Unable to find file HELP.HLP.$'));
485  3            end;
486  2            call movef(13,.(0,'HELP    DAT',0),.fcb2);
487  2            call delete$file(.fcb2);
488  2            if make$file(.fcb2) = 0FFH then
489  2            do;
490  3               i = close$file(.fcb);
491  3               call error(true,.('Unable to Make HELP.DAT.$'));
492  3            end;
493  2            call set$dma(.sysbuff);
494  2            end$index = 0;
495  2            do while ((i != read$record(.fcb)) = 0) and (not end$index);
496  3                 if sysbuff(7).subject(0) = '$' then end$index = true;
498  3            end;
499  2            eod = 0;
500  2            if i <> 0 then eod = true;
502  2            i = write$record(.fcb2);
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #

```
503  2          save$size = maxb - .memory;
504  2          do while (not eod);
505  3             count = 0;
506  3             max$buf = save$size;
507  3             do while (not eod) and (max$buf > 127);
508  4                call set$dma(.memory(shl(count,7)));
509  4                if read$record(.fcb) <> 0 then
510  4                do;
511  5                   eod = true;
512  5                end;
                      else
513  4                do;
514  5                   max$buf = max$buf - 128;
515  5                   count = count + 1;
516  5                end;
517  4             end;
518  3             do count2 = 0 to count-1;
519  4                call set$dma(.memory(shl(count2,7)));
520  4                if write$record(.fcb2) = 0FFH then
521  4                do;
522  5                   i = close$file(.fcb2);
523  5                   call delete$file(.fcb2);
524  5                   i = close$file(.fcb);
525  5                   call error(true,.('Writing file HELP.DAT.$'));
526  5                end;
527  4             end;
528  3          end;
529  2          if close$file(.fcb) = 0FFH then
530  2          do;
531  3             call error(false,.('Unable to Close HELP.HLP.$'));
532  3          end;
533  2          if close$file(.fcb2) = 0FFH then
534  2          do;
535  3             call delete$file(.fcb2);
536  3             call error(true,.('Unable to Close HELP.DAT.$'));
537  3          end;
538  2          call print$console$buf(.('Extraction complete',cr,lf,lf,
                                        'HELP.DAT created',cr,lf,'$'));

539  2       end extract$file;

          /*************************************************
                    Display$ind Procedure

               Displays the avialable topics
          *************************************************/
540  1       display$ind:
            procedure;
541  2          declare (disp$level,i,eod,written) byte;
542  2          declare (offset,index,count) address;
543  2          declare name (14) byte;
544  2           offset,
              written,
              eod = 0;
545  2          disp$level = level + 1;
546  2          if disp$level < 10 then
547  2          do;
```

COPYRIGHT ©1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950
SER. #_____

```
548  3            if level = 0 then
549  3            do;
550  4               offset = 0;
551  4            end;
                  else
552  3            do;
553  4               offset = gindex;
554  4            end;
555  3            count = 0;
556  3         end;
               else
557  2         do;
558  3            eod = true;
559  3         end;
560  2         index = offset;
561  2         offset = 0;
562  2         do while (not eod);
563  3              if sysbuff(index).subject(0) = '$' then
564  3              do;
565  4                 eod = true;
566  4              end;
                    else
567  3              do;
568  4                 if sysbuff(index).level = disp$level then
569  4                 do;
570  5                    if not written then
571  5                    do;
572  6                       written = true;
573  6                       i = page$check(.tcnt);
574  6                       if disp$level = 1 then
575  6                       do;
576  7                          call print$console$buf(.(cr,'Topics available:$'));
577  7                       end;
                             else
578  6                       do;
579  7                          call print$console$buf(.(cr,'ENTER .subtopic FOR ',
                                        'INFORMATION ON THE FOLLOWING SUBTOPICS:$'));
580  7                       end;
581  6                       i = page$check(.tcnt);
582  6                       call print$console$buf(.(cr,'$'));
583  6                    end;
584  5                    if (count mod display$cols) = 0 then
585  5                    do;
586  6                       i = page$check(.tcnt);
587  6                       call write$console(cr);
588  6                    end;
589  5                    do i = 0 to 13;
590  6                       name(i) = ' ';
591  6                    end;
592  5                    name(13) = '$';
593  5                    call movef(12,.sysbuff(index).subject,.name);
594  5                    call print$console$buf(.name);
595  5                    count = count + 1;
596  5                 end;
                       else
597  4                 do;
598  5                    if sysbuff(index).level < disp$level then eod = true;
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950
SER. #_____

```
600  5                end;
601  4                index = index + 1;
602  4              end;
603  3          end;
604  2          if written then
605  2            do;
606  3               i = page$check(.tcnt);
607  3               call print$console$buf(.(cr,lf,'$'));
608  3            end;
609  2          call set$dma(.tbuff);
610  2       end display$ind;

          /*********************************************
                  Search$file Procedure

              Searches the index table for the key
          *********************************************/
611  1       search$file:
            procedure byte;
612  2          declare (eod, error, cnt, found, saved, save$level) byte;
613  2          declare index address;
614  2          eod,
              error,
              found,
              saved,
              index = 0;
615  2          do while(not eod) and (not error);
616  3               if sysbuff(index).subject(0) <> '$' then
617  3               do;
618  4                  if sysbuff(index).level = level + 1 then
619  4                  do;
620  5                     cnt = compare(.com(level).name,.sysbuff(index).subject);
621  5                     if cnt = 0 then
622  5                     do;
623  6                        call movef(12,.sysbuff(index).subject,.com(level).name);
624  6                        level = level + 1;
625  6                        if (not saved) then
626  6                        do;
627  7                           save$level = level;
628  7                           saved = true;
629  7                        end;
630  6                        if ((level > 8) or (com(level).name(0) = ' '))
                                        or (com(level).name(0) = '[') then
631  6                        do;
632  7                           found = true;
633  7                           eod = true;
634  7                        end;
                              else
635  6                        do;
636  7                           index = index + 1;
637  7                           found = 0;
638  7                        end;
639  6                     end;
                           else
640  5                     do;
641  6                        index = index + 1;
642  6                     end;
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950
SER. #_____

```
643  5                end;
                      else
644  4                do;
645  5                   if saved then
646  5                   do;
647  6                      if save$level < sysbuff(index).level then
648  6                      do;
649  7                         index = index + 1;
650  7                      end;
                            else
651  6                      do;
652  7                         error = true;
653  7                      end;
654  6                   end;
                         else
655  5                   do;
656  6                      index = index + 1;
657  6                   end;
658  5                end;
659  4             end;
                   else
660  3             do;
661  4                error = true;
662  4             end;
663  3          end;
664  2          if found then
665  2          do;
666  3             gindex = index + 1;
667  3             call movef(1,.sysbuff(index).record,.fcb(33));
668  3             call movef(1,.sysbuff(index).record+1,.fcb(34));
669  3             fcb(35) = 0;
670  3             offset = sysbuff(index).rec$offset;
671  3             level = sysbuff(index).level;
672  3          end;
673  2          return error;
674  2       end search$file;

           /**************************************
                  Token Display Procedure

                Displays the Parsed Tokens
           **************************************/
675  1       display$tokens:
             procedure (no$tokens);
676  2          declare (token$cnt1, token$cnt2, no$tokens) byte;
677  2          token$cnt1 = 0;
678  2          do while (token$cnt1 < no$tokens) and (not eod);
679  3             eod = page$check(.tcnt);
680  3             if (not eod) then
681  3             do;
682  4                do token$cnt2 = 0 to token$cnt1;
683  5                   call print$console$buf(.(' $'));
684  5                end;
685  4                call print$console$buf(.com(token$cnt1).name);
686  4                token$cnt1 = token$cnt1 + 1;
687  4             end;
688  3          end;
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #

```
689  2       end display$tokens;

             /**************************************
                       Print Procedure

                  Displays the Help text
             **************************************/
690  1       print:
               procedure;
691  2         declare (i,ii,char,eod2) byte;
692  2         declare temp(3) byte;
693  2         call write$console(cr);
694  2         call display$tokens(level);
695  2         if (not eod) then eod = page$check(.tcnt);
697  2         if (not eod) then
698  2         do;
699  3            if read$rand(.fcb) <> 0 then
700  3            do;
701  4               offset =close$file(.fcb);
702  4               call error(true,.('Reading file HELP.HLP,$'));
703  4            end;
                  else
704  3            do;
705  4               eod2 = 0;
706  4               do while ((not eod2) and (not eod)) and (read$record (.fcb) = 0);
707  5                  i = offset - 1;
708  5                  do while (((i:=i+1) <= 127) and (not eod2));
709  6                     if (char := tbuff(i)) = control$z then eod = true;
711  6                     ii = 0;
712  6                     do while((not eod2) and (not eod)) and
                                                ((ii < 3) and (tbuff(i) = slash));
713  7                        ii = ii + 1;
714  7                        i = inc(i);
715  7                        temp(ii-1) = tbuff(i);
716  7                     end;
717  6                     if ii = 3 then eod2 = true; else temp(ii) = '$';
720  6                     if ((not eod) and (not eod2)) then
721  6                     do;
722  7                        if (char = lf) and (not nopage$mode) then
723  7                        do;
724  8                           eod = page$check(.tcnt);
725  8                        end;
                              else
726  7                        do;
727  8                           call write$console (char);
728  8                        end;
729  7                        if ii > 0 then call print$console$buf(.temp);
731  7                        ii = 0;
732  7                     end;
733  6                  end;
734  5                  offset = 0;
735  5               end;
736  4            end;
737  3         end;
738  2         eod = 0;
739  2       end print;
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #_____

```
                /**************************************
                        Prompt Procedure

                  Prompts for input from the user
                **************************************/
740   1         prompt:
                  procedure byte;
741   2           declare temp byte;
742   2           call movef(1,.(128),.tbuff-1);
743   2           temp = page$check(.tcnt);
744   2           call print$console$buf(.(cr,'HELP> $'));
745   2           call read$console$buff(.tbuff-1);
746   2           tbuff(tbuff(0)+1) = 0;
747   2           tcnt = -1;
748   2           temp = parse;
749   2           if (temp <> 0) and (not list$mode)
                    then call print$console$buf(.clear$screen);
751   2           return temp;
752   2         end prompt;


                /**************************************
                        Main Program
                **************************************/

753   1         declare last$dseg$byte byte
                  initial (0);


754   1         plm:
                  do;
755   2           eod,
                    tcnt = 0;
756   2           version = get$version;
757   2           if (high(version) = 1) or (low(version) < 30h) then
758   2             do;
759   3               call error(true,.('Requires CP/M Version 3$'));
760   3             end;
761   2           page$len = mon2(49,.(1ch,0,)) - 1;
762   2           display$cols = low((mon2(49,.(1ah,0))+1) / 13);
763   2           if mon2(49,.(2ch,0)) = 0 then
764   2             page$mode = true;
                    else
765   2             page$mode = false;
766   2           cnt = parse;
767   2           if create$mode then
768   2           do;
769   3             call create$index;
770   3           end;
                    else
771   2           if extract$mode then
772   2           do;
773   3             call extract$file;
774   3           end;
                    else
775   2           do;
776   3             call movef(13,.(0,'HELP ',0A0H,'  HLP',0),.fcb); /* open read/only */
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950

SER. #_____

```
777  3          if open$file (.fcb) <> 0FFH then
778  3          do;
779  4            call init;
780  4            if (not list$mode) then
781  4              call print$console$buf(.clear$screen);
782  4            if cnt = 0 then
783  4            do;
784  5               level = 0;
785  5               call print$console$buf(.(cr,lf,'HELP UTILITY V1.1',cr,lf,lf,
                                              'At "HELP>" enter ',
                                              'topic {,subtopic}...',cr,lf,lf,
                                              'EXAMPLE:  HELP> DIR EXAMPLES',
                                               cr,lf,'$'));
786  5               tcnt = 2;
787  5               call display$ind;
788  5               cnt = prompt;     /* Prompt for user input */
789  5            end;
790  4            do while cnt <> 0;   /* If user didn't hit a return do */
791  5             level = 0;
792  5             if compare(.com(0).name,.('?          ')) = 0 then
793  5             do;
794  6               ; /* NULL COMMAND */
795  6             end;
                   else
796  5             if search$file <> 0FFH then
797  5             do;
798  6                call print;
799  6                if compare(.com(0).name,.('HELP       ')) = 0 then
800  6                do;
801  7                   level = 0;
802  7                end;
803  6             end;
                   else
804  5             do;
805  6                eod = page$check(.tcnt);
806  6                call write$console(cr);
807  6                if (not eod) then
808  6                do;
809  7                   eod = page$check(.tcnt);
810  7                   if (not eod) then
811  7                   do;
812  8                      call print$console$buf(.('Topic:$'));
813  8                      eod = page$check(.tcnt);
814  8                      call write$console(cr);
815  8                      call display$tokens(cnt);
816  8                      eod = page$check(.tcnt);
817  8                      call write$console(cr);
818  8                      eod = page$check(.tcnt);
819  8                      call write$console(cr);
820  8                      call print$console$buf(.('Not found$'));
821  8                      eod = page$check(.tcnt);
822  8                      call write$console(cr);
823  8                   end;
824  7                end;
825  6                level = 0;
826  6             end;
827  5             if (not eod) then call display$ind;
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #

```
 829   5              cnt = prompt;   /* Prompt for user input */
 830   5            end;
 831   4           offset = close$file(.fcb);
 832   4         end;
                  else
 833   3         do;
 834   4           call error(false,.('No HELP.HLP file on the default drive.$'));
 835   4         end;
 836   3       end;
 837   2       end;
 838   1       call terminate;
 839   1      end help;
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #_____

MODULE INFORMATION:

```
     CODE AREA SIZE     = 18E3H   6371D
     VARIABLE AREA SIZE = 01AAH    426D
     MAXIMUM STACK SIZE = 000AH     10D
     1089 LINES READ
     0 PROGRAM ERROR(S)
```

END OF PL/M-80 COMPILATION

```
0000 HELP
0000 MCD80A
0004 BDISK      008C BUFF       0080 BUFFA      0050 CMDRV      0000 CPU
007C CR         006D DOLLA      005C FCB        006C FCB16      005C FCBA
005C IFCB       005C IFCBA      0003 IOBYTE     0053 LEN0       0056 LEN1
0006 MAXB       0006 MEMSIZ     0005 MON1       0005 MON2       0005 MON2A
0005 MON3       0000 OFFSET     006E PARMA      0051 PASS0      0054 PASS1
007F RO         007D RR         007D RRECA      007F RRECO      005C SFCB
0080 TBUFF
0000 HELP
1CE9 MEMORY     0584 PLM        1B3F LISTMODE   1B40 NOPAGEMODE 1B41 CREATEMODE
1B42 EXTRACTMODE                1B43 PAGEMODE   1B44 OFFSET     1B45 EOD
1B46 FCB2       1B6A CNT        1B6B INDEX      1B6C SUB        1B78 COM
1CE9 SYSBUFF    1C1D NAME       1C29 LEVEL      1C2A GINDEX     1C2C TCNT
1C2D VERSION    1C2F PAGELEN    1C30 DISPLAYCOLS
1C31 CLEARSCREEN                073F READCONSOLE
0748 WRITECONSOLE               1C4B CHAR       0758 PRINTCONSOLEBUF
1C4C BUFFADR    0768 READCONSOLEBUFF            1C4E BUFFADR
0778 DIRECTCONIO                1C50 FUNC       0788 GETVERSION 0791 DELETEFILE
1C51 FCBADDRESS 07A1 OPENFILE   1C53 FCBADDRESS 07C3 CLOSEFILE  1C55 FCBADDRESS
07D3 READRECORD 1C57 FCBADDRESS 07E3 WRITERECORD                1C59 FCBADDRESS
07F3 MAKEFILE   1C5B FCBADDRESS 0815 READRAND   1C5D FCBADDRESS 0825 SETDMA
1C5F DMAADDRESS 0835 SETRANDREC 1C61 FCBADDRESS 0845 TERMINATE  084E ERROR
1C63 TERMCODE   1C64 ERRMSGADR  0875 MOVEF      1C66 MVCNT      1C67 SOURCEADDR
1C69 DESTADDR   089B COMPARE    1C6B STR1ADDR   1C6D STR2ADDR   1C6F RESULT
1C70 I          0921 INC        1C71 INCI       0956 PAGECHECK
1C72 LINECNTADDR                1C74 QUIT       09E9 INIT       1C75 BUFSIZE
1C77 MAXBUF     1C79 INITI      1C7B ENDINDEX   0AB7 PARSE      1C7C INDEX
1C7D BEGIN      1C7E CNT        1C7F I          1C80 STOP       1C81 BRACKET
0E2B CREATEINDEX                1C82 CNT        1C83 I          1C84 RECCNT
1C85 INDEX      1C87 COUNT      1C89 COUNT2     1C8B MAXBUF     1C8D SAVESIZE
1C8F FCB3       134A EXTRACTFILE                1CB3 ENDINDEX   1CB4 I
1CB5 COUNT      1CB7 COUNT2     1CB9 MAXBUF     1CBB SAVESIZE   14F0 DISPLAYIND
1CBD DISPLEVEL  1CBE I          1CBF EOD        1CC0 WRITTEN    1CC1 OFFSET
1CC3 INDEX      1CC5 COUNT      1CC7 NAME       1664 SEARCHFILE 1CD5 EOD
1CD6 ERROR      1CD7 CNT        1CD8 FOUND      1CD9 SAVED      1CDA SAVELEVEL
1CDB INDEX      182E DISPLAYTOKENS              1CDD NOTOKENS   1CDE TOKENCNT1
1CDF TOKENCNT2  189A PRINT      1CE0 I          1CE1 II         1CE2 CHAR
1CE3 EOD2       1CE4 TEMP       1A0E PROMPT     1CE7 TEMP
1CE8 LASTDSEGBYTE
```

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950

SER. #_____

```
0000 HELP$
0000 MCD80A$
0000 HELP$
```

073F 41	073F 42	0748 43	0748 44	074C 46
0757 47	0758 48	075E 50	0767 51	0768 52
076E 54	0777 55	0778 56	077C 58	0788 59
0788 60	0788 61	0791 62	0791 63	0797 65
07A0 66	07A1 67	07A7 70	07B0 71	07B9 72
07C3 73	07C3 74	07C9 76	07D3 77	07D3 78
07D9 80	07E3 81	07E3 82	07E9 84	07F3 85
07F3 86	07F9 89	0802 90	080B 91	0815 92
0815 93	081B 95	0825 96	0825 97	082B 99
0834 100	0835 101	083B 103	0844 104	0845 105
0845 106	084D 107	084E 108	0856 111	085C 112
0864 113	086A 114	0871 115	0874 116	0875 117
0884 120	089A 121	089B 122	089B 123	08A5 128
08AD 129	08CB 130	08E6 132	0901 134	0906 135
0909 137	090E 138	090E 139	0913 140	0913 141
091A 142	091D 143	0921 144	0921 145	0925 147
092C 148	0935 150	0940 152	0945 153	0948 155
094D 156	0952 157	0952 158	0952 159	0956 160
0956 161	095C 165	0961 166	096D 168	097E 170
0984 171	0989 172	0992 173	099B 174	099E 175
09A4 176	09AD 178	09B7 179	09BA 180	09BD 182
09C6 184	09CB 185	09CB 186	09D0 187	09D0 188
09D3 190	09D8 191	09D8 192	09DB 194	09E0 195
09E5 196	09E5 197	09E9 198	09E9 199	09E9 202
09F5 203	09FB 204	0A00 205	0A06 206	0A1B 207
0A30 208	0A3B 210	0A47 211	0A4F 212	0A4F 213
0A60 214	0A65 215	0A6D 217	0A7A 218	0A84 219
0A84 220	0A87 221	0A8D 222	0AA2 224	0AAE 225
0AB6 226	0AB6 227	0AB7 228	0AB7 230	0ABC 231
0AC4 233	0AD3 234	0AE2 235	0AF0 236	0AFF 237
0B0A 238	0B14 239	0B19 240	0B37 241	0B3E 242
0B41 243	0B50 245	0B56 246	0B61 247	0B64 248
0B69 249	0B78 250	0B91 251	0B9B 252	0BA1 253
0BA6 254	0BAE 255	0BC9 256	0BD8 258	0BDD 259
0C27 260	0C46 262	0C6D 263	0C70 265	0C94 266
0C94 267	0C9B 268	0CA2 269	0CA5 270	0CAC 271
0CBB 273	0CC0 274	0CC3 275	0CD2 277	0CE9 278
0CF0 279	0D01 280	0D08 281	0D0F 282	0D14 283
0D14 284	0D17 286	0D1E 287	0D1E 288	0D21 289
0D21 290	0D2F 291	0D38 293	0D3D 294	0D45 295
0D5A 297	0D6F 299	0D74 300	0D7C 301	0D7F 302
0D94 304	0D99 305	0DA1 306	0DA4 307	0DB9 309
0DBE 310	0DC6 311	0DC9 312	0DDE 314	0DE3 315
0DE6 316	0DEE 317	0DF1 318	0E06 320	0E0E 321
0E11 323	0E18 324	0E18 325	0E1D 326	0E1D 327
0E24 328	0E27 329	0E27 330	0E2B 331	0E2B 332
0E2B 336	0E31 337	0E3F 338	0E56 339	0E60 340
0E6C 341	0E78 342	0E7E 343	0E8A 344	0E95 346
0E9D 347	0E9D 348	0EA2 349	0EB9 350	0EBE 351
0ED0 352	0EDF 354	0EE4 355	0EE7 357	0EF6 359
0EFB 360	0F16 361	0F20 362	0F27 363	0F2A 364
0F3E 366	0F5A 367	0F64 368	0F69 369	0F8E 370
0FAD 372	0FCD 373	0FD0 375	0FF0 376	0FF0 377
0FFA 378	1001 379	1004 380	100C 382	1012 383
102B 384	1045 385	106C 386	107F 387	1086 388
1099 390	10A0 391	10AD 392	10C2 394	10CB 395
10D3 396	10D6 397	10ED 398	1102 399	110F 400
110F 401	110F 402	110F 403	1112 405	111C 406
111C 407	111C 408	111F 409	1122 410	1128 411
112F 412	113B 413	1141 414	114C 416	1155 417

COPYRIGHT © 1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950

SER. #_____

115B 418 1164 419 116C 420 116C 421 1178 422
1181 423 1195 424 11BE 425 11CB 426 11DE 427
11EF 428 11FA 430 1203 431 1209 432 1212 433
1218 434 1221 435 1229 436 1229 437 1236 438
1242 439 124B 440 1250 441 1258 442 125E 443
1264 444 1279 445 128A 446 1295 448 129A 449
129D 451 12AA 452 12B1 453 12B1 454 12B4 455
12C8 456 12D9 457 12E4 459 12ED 460 12F3 461
12FC 462 1304 463 1304 464 1311 465 1314 466
131F 468 1328 469 1330 470 1330 471 133B 473
1343 474 1343 475 1349 476 134A 477 134A 480
1350 481 135C 482 1367 484 136F 485 136F 486
137B 487 1381 488 138C 490 1395 491 139D 492
139D 493 13A3 494 13A8 495 13C2 496 13CA 497
13CF 498 13D2 499 13D7 500 13DF 501 13E4 502
13ED 503 13F9 504 1401 505 1407 506 140D 507
1422 508 1433 509 143E 511 1443 512 1446 514
1453 515 145A 516 145A 517 145D 518 1471 519
1482 520 148D 522 1496 523 149C 524 14A5 525
14AD 526 14AD 527 14BA 528 14BD 529 14C8 531
14D0 532 14D0 533 14DB 535 14E1 536 14E9 537
14E9 538 14EF 539 14F0 540 14F0 544 14FD 545
1504 546 150C 548 1514 550 151A 551 151D 553
1523 554 1523 555 1529 556 152C 558 1531 559
1531 560 1537 561 153D 562 1545 563 1556 565
155B 566 155E 568 1574 570 157C 572 1581 573
158A 574 1592 576 1598 577 159B 579 15A1 580
15A1 581 15AA 582 15B0 583 15B0 584 15C5 586
15CE 587 15D3 588 15D3 589 15E1 590 15EC 591
15F6 592 15FB 593 1611 594 1617 595 161E 596
1621 598 1638 599 163D 600 163D 601 1644 602
1644 603 1647 604 164E 606 1657 607 165D 608
165D 609 1663 610 1664 611 1664 614 167A 615
168A 616 169B 618 16B2 620 16D5 621 16DD 623
1700 624 1707 625 170F 627 1715 628 171A 629
171A 630 1755 632 175A 633 175F 634 1762 636
1769 637 176E 638 176E 639 1771 641 1778 642
1778 643 177B 645 1782 647 1798 649 179F 650
17A2 652 17A7 653 17A7 654 17AA 656 17B1 657
17B1 658 17B1 659 17B4 661 17B9 662 17B9 663
17BC 664 17C3 666 17CA 667 17E4 668 17FF 669
1804 670 1817 671 182A 672 182A 673 182E 674
182E 675 1832 677 1837 678 184B 679 1854 680
185C 682 186B 683 1871 684 187B 685 188F 686
1896 687 1896 688 1899 689 189A 690 189A 693
189F 694 18A6 695 18AE 696 18B7 697 18BF 699
18CA 701 18D3 702 18DB 703 18DE 705 18E3 706
1902 707 1909 708 1922 709 1934 710 1939 711
193E 712 196B 713 1972 714 197C 715 1994 716
1997 717 199F 718 19A7 719 19B2 720 19C2 722
19D6 724 19DF 725 19E2 727 19E9 728 19E9 729
19F2 730 19F8 731 19FD 732 19FD 733 1A00 734
1A05 735 1A08 736 1A08 737 1A08 738 1A0D 739
1A0E 740 1A0E 742 1A1A 743 1A23 744 1A29 745
1A2F 746 1A3A 747 1A3F 748 1A45 749 1A59 750
1A5F 751 1A63 752 0581 754 0587 755 0591 756
0597 757 05AF 759 05B7 760 05B7 761 05C3 762
05D9 763 05E6 764 05EE 765 05F3 766 05F9 767
0600 769 0603 770 0606 771 060D 773 0610 774
0613 776 061F 777 062A 779 062D 780 0635 781
063B 782 0643 784 0648 785 064E 786 0653 787
0656 788 065C 789 065C 790 0664 791 0669 792
0677 795 067A 796 0682 798 0685 799 0693 801
0698 802 0698 803 069B 805 06A4 806 06A9 807
06B1 809 06BA 810 06C2 812 06C8 813 06D1 814
06D6 815 06DD 816 06E6 817 06EB 818 06F4 819

COPYRIGHT ©1982
DIGITAL RESEARCH
P. O. BOX 579
PACIFIC GROVE, CA 93950
SER. #_____

06F9 820 06FF 821 0708 822 070D 823 070D 824
070D 825 0712 826 0712 827 071A 828 071D 829
0723 830 0726 831 072F 832 0732 834 073A 835
073A 836 073A 838 073D 839

COPYRIGHT © 1982
DIGITAL RESEARCH
P.O. BOX 579
PACIFIC GROVE, CA 93950

SER. #_____ ____